1. Baggrund

Databasen der ligger grundlaget for Samaquas ArcGis har sin data fra mange forskellige kilder. Dataet er blevet indsat på flere forskellige måder, f.eks; Noget af dataet er manuelt indskrevet, mens andet er konverteret fra ældre CAD filer. På grund af dette, er der opstået datafejl i databasen, med tabeller der ikke er udfyldt korrekt, til shapefiler der ikke er korrekt placeret. Eventuelle fejl der allerede eksisterede i CAD filerne er også blevet arvet. Samaqua ønsker nogle scripts som kan finde disse fejl og indsætte dem i en fejllog, så de kan efterses og rettes.

1. Problemstilling

På grund af de mange forskellige måder hvorpå fejl kan gemme sig i databasen, kræves der flere scripts til at finde enkelte typer fejl. Da metoderne som bruges til at udvælge fejlene, er påfund af forskellige mennesker, skal fejlfinderen kunne udvides som flere udvælgelses metoder bliver fundet. Loggen som bruges til at få et overblik over mængden af fejl skal også være fleksibel nok til at den kan håndtere nye fejltyper som findes i fremtiden.

Da mængden af scripts der fejlfinder kan risikere at blive stor, vil det være en fordel hvis hvert enkelt script ikke har en for stor køretid, da køringen af flere af dem på en gang derved kan kræve for meget tid. Scriptende skal også være uafhængige af hinanden, i tilfældet af at der kun ønskes at køre nogle af dem eller køre dem I en uspecifik rækkefølge.

1. Løsning

De forskellige scripts finder deres specifikke fejl på forskellige måder, men de fleste af dem finder fejl ved hjælp af cursors på databasen, joins og SQL selections. Hvert script gemmer derefter de fejl de finder I deres specifikke fejldatabase, og skriver til fejlloggen sin fejltype og hvor mange fejl der blev fundet. Herunder er en tabel over de Scripts der er skrevet hidtil, og en kort beskrivelse af dem. For mere information, se deres sourcecode.

|  |  |  |  |
| --- | --- | --- | --- |
| Script Navn | Tabeller der fejlsøges | Script Beskrivelse | Felter der gemmes |
| 1 KnudeNamelen | Node | Finder de brønde med navne længere end 30 tegn | ObjectID  NodeName |
| 2 NodevalidRefrence | Node  ComplexStructure | Finder de brønde som refererer til et komplekst bygningsværk som ikke findes i den tabel | Node.ObjectID  Node.StructureRefID |
| 3 LateralvalidRefrence | LateralConnection  MainPipe | Finder de stik som refererer til et afløb der ikke findes i den tabel | Lateral.ObjectID  Lateral.MainPipeObjectID |
| 4 NodeMultiplecover | Cover | Finder brønde som har mere end et dæksel som referer den | ObjectID  NodeRefID |
| 5 MainPipeMissingPipes | MainPipe  Pipe | Finder de afløbs hovedledninger som ikke har nogle delledninger | MainPipe.ObjectID  MainPipe.UpstreamNode  MainPipe.DownstreamNode |
| 6 OwnerMatch | MainPipe  Owner | Finder de afløb hvis ejerkode ikke matcher deres beskrevet ejer | MainPipe.ObjectID  MainPipe.OwnerID  MainPipe.OwnerCode |
| 7 MainPipeFlowConflict | MainPipe  Pipe | Finder de afløb hvis delledninger ikke går fra start til slutbrønden. | MainPipe.ObjectID  Pipe.ObjectID  Node.ObjectID |
| 8 ReportClockRefrence | CCTVInspection  Report  ReportFromToward  Node  LateralConnection | Finder de rapporter hvis stiks urreference ikke matcher stikkets eget urreference | CCTVInspection.ObjectID  Report.ObjectID  Report.ReportNr  Node.ObjectID  Node.NodeName |
| 9 CoverNodeCoordinates | Cover  Node | Finder de dæksler hvis koordinater er over 1m væk fra sin tilhørende brønd | Cover.ObjectID  Node.ObjectID  \*Distance from Shapefiles |
| 10 PipeNodeCoordinates | Pipe  Node | Finder de delledninger som ikke er forbundet deres knuder | Pipe.ObjectID  Node.ObjectID  \*Distance from Shapefiles |
| 11 ShapeFieldCoordinates | Node | Finder de knuder hvis tabel og shapefil koordinater ikke matcher | ObjectID  \*Distance between Shapefile and Table value |
| 12 LateralPipeCoordinates | MainPipe  Pipe  Node  LateralConnection | Finder de stik hvis shapefil ikke er forbundet deres afløbs shapefil | MainPipe.ObjectID  Node.ObjectID  \*Distance from Shapefiles |
| 13 TVInspectionvalidRefrence | CCTVInspection  Report  PipeReport | Finder de inspektioner der ikke har en valid reference til et afløb | CCTVInspection.ObjectID  CCTVInspection .InActive |
| 14 CovervalidNode | Cover  Node | Finder de dæksler som er på en knude med en typekode der ikke burde have et dæksel | Cover.ObjectID  Node.ObjectID  Node.NodeTypeCode |
| 15 Nodetooclose | Node | Finder noder der er for tæt på hinanden (inden for 2 cm) | ObjectID  NodeTypeCode |
| 16 NodetoomanyPipes | Pipe  Node | Finder knuder der har, men ikke må have mere end 2 delledninger på sig | Node.ObjectID  Node.NodeTypeCode |
| 17 PipedepthNode | Pipe  Node | Finder delledninger som er forbundet på en knude over dens terrænkvote eller under bundkvoten. | Pipe.ObjectID  Pipe.InvertLevel  Node.ObjectID  Node.InvertLevel  Node.GroundLevel |

Mængden af fejl af de forskellige fejltyper gemmes også i sin egen Error Log, der kan bruges til hurtigt at få et overblik over fejlene. Derudover gemmer loggen også datoen for fejlfindingen som en Timestamp, og gemmer en Run ID som man kan se hvilket fejl der er fundet i samme køring.

1. Appendix

Eksempel på en tabel som et script skriver sine fundne fejl til.

*\* PATH er stien til placeringen af tabellen*

dbname = "DB\_ErrTypeX"

arcpy.CreateTable\_management(PATH\*,dbname )

arcpy.AddField\_management(dbname ,"DetectionInfo","TEXT")

arcpy.AddField\_management(dbname ,"GisLocationInfo","TEXT")

arcpy.AddField\_management(dbname ,"Note","TEXT")

arcpy.AddField\_management(dbname ,"ErrID","LONG")

arcpy.AddField\_management(dbname ,"PipeObjID","LONG")

arcpy.AddField\_management(dbname ,"NodeObjID","LONG")

arcpy.AddField\_management(dbname ,"Distance","FLOAT")